CLAWAR 2019: 22nd International Conference on
Climbing and Walking Robots and the Support
Technologies for Mobile Machines,
Kuala Lumpur, Malaysia, 26-28 August 2019.
https://doi.org/10.13180/clawar.2019.26-28.08.35

CONTROL OF A TWO-WHEELED LEGO EV3 ROBOT USING INTERVAL TYPE-2 FUZZY LOGIC WITH PARTICLE SWARM OPTIMIZATION

W.S.KIN, N.M.A GHANI, M.F.MASROM, N.F. JAMIN and N.A.A RAZALI
Faculty of Electrical & Electronic Engineering, Universiti Malaysia Pahang, 26600 Pekan Pahang
E-mail: normaniha@ump.edu.my
www.ump.edu.my

Two-wheeled robot self-balancing has gained much interest of researchers due to its nonlinear dynamics. This project is aimed to design an Interval Type-2 Fuzzy Logic Controller to control a two-wheeled LEGO EV3 robot self-balancing to keep it in the upright position. In this project, two-wheeled LEGO EV3 robot is modelled using SimWise 4D software and integrated with Simulink. The robot stability performance and output response are observed at the same time when the Simulink is executed. System identification is used to get the mathematical model of the system in state space based on input and output from SimWise 4D motion to compare both results. The state space is used during optimization of IT2FLS using Particle Swarm Optimization (PSO). The performances of Interval Type 2 Fuzzy Logic Controller (IT2FLC) and optimized IT2FLS are compared. The robustness of IT2FLS is observed during disturbance rejection by injecting different direction of 0.8N and 1.0N torque to the robot in first 15 seconds.

1. Introduction

Since fuzzy sets were first established by L. A. Zadeh in 1965 and rapid research on the fuzzy theory application, the fuzzy control becomes famous with its features of multi-variables, high order, nonlinear, and strong coupling. It performs better in classification and control. A self-balancing robot is one of the common platforms of research to exemplify system stability, controllability, anti-interference, and robustness [1-6]. Fuzzy logic control can reflect the human thinking and convert probability theory into mathematical logic. However, it does not require a mathematical model for implement the control function.

The control system of a two-wheeled self-balancing robot has been studied for many years. There are various types of control algorithms which always being used in self-balancing, like LQR, LQG, PID control, pole placement, adaptive control, fuzzy logic control and so on [7-15]. However, there are still some open issues on the efficiency of different types of control. The majority controls that are being used in the industry nowadays are more to linear control. In [4], the implementation of a pole-placement controller in two-wheeled self-balancing robot equipped with supporting arms maintains robot upright in range of ±30˚. The comparison of PID and LQR controller in [6] also shows the highly unsuitable for nonlinearities in PID controller due to overshoots and LQR performs long rise time in self-balancing the two-wheeled robot.

In [16], the PSO-based Fuzzy control shows great control ability in maintaining system stability. In [17], the Fuzzy PD controller also improves the balance of robot from ±45˚ to ±10˚. There is a lot of application of FLC in common household devices currently. The advantages of using FLC in these applications include automatic compensation for operator injected noise in camcorders, making intelligent floor decisions and minimizing travel and power consumption [18-20]. There is a big gap between application and theory. In industrial operation, most of the operators using linear controllers instead of Type-2 Fuzzy Logic Control due to its simple structure and easy implementation. Plus, the Type-2 Fuzzy

Controller is quite heavy compared with Fuzzy Logic Controller. However, Type-2 Fuzzy Control can provide additional design degrees of freedom in the fuzzy logic system [21-22].

Fuzzy logic systems have been credited in robotics control and capable provide robust control for modelling system that subjected to uncertainties [23]. The flexible set of if-then rules in FLC driving force in increasing the popularity of Fuzzy logic control. FLC is able to quantify the input signal in a noisy environment and apply smart fit data. However, the general fuzzy logic approach is difficult to overcome the uncertainties that exist in a large number of the real application with dynamical environments. It is unable to model knowledge adequately but Type-2 fuzzy logic can offer a high level of imprecision [24-30].

2. System Model and Parameters

2.1. *Modelling of a two-wheeled EV3 Lego Robot*

The LEGO EV3 robot is being modelled using SimWise 4D. The parameters, descriptions, values, and units are tabulated in Table 1. Gain *K1* is the translation kinetic energy, gain *K2* is the rotation kinetic energy, and U is the potential energy. Assume θ to be the angular position of the left and right wheel, ψ to be the tilt angle and φ to be the yaw angle.

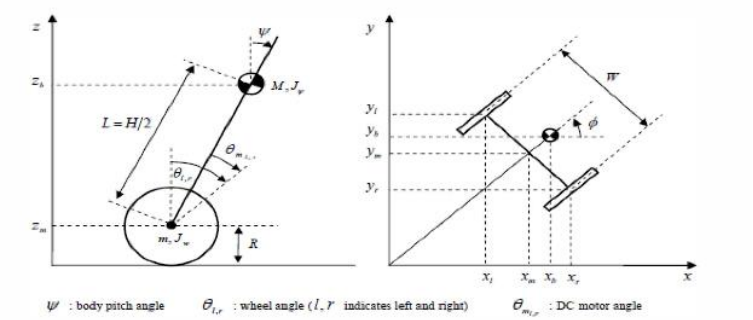

Figure 1. Inverted two-wheeled pendulum [2], [32]

Table 1. Parameters of Lego Ev3 Robot [18].

J_w	mR^2/L	[kgm^2]	Wheel inertia moments
M	0.610	[kg]	Body weight
W	0.156	[m]	Body width
D	0.050	[m]	Body depth
H	0.250	[m]	Body height
L	H/2	[m]	Distance of the center of mass from wheel axle
J_φ	M L^2/3	[kgm^2]	Body pitch inertia moment
J_m	0.00001	[kgm^2]	DC motor inertia
R_m	6.690	[Ω]	DC motor resistance
K_b	0.468	[Vsec/rad]	DC motor back emf constant
K_t	0.317	[Nm/A]	DC motor torque constant
n	1		Gear ratio
f_m	0.0022		Friction coefficient between body and DC motor
f_w	0		Friction coefficient between wheel and motor
g	9.810	[m/sec^2]	Gravity constant
m	0.022	[kg]	Mass of wheels
R	0.029	[m]	Radius of wheels

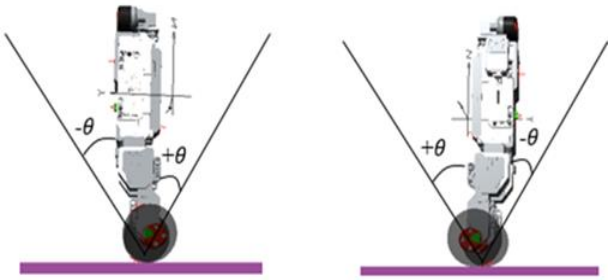

Figure 2. The angular position of left and right wheels.

The initial tilt angle of LEGO EV3 robot (gyro) is zero radian.

Figure 3. The gyro meter of orientation in SimWise 4D.

The LEGO EV3 robot is completed by connecting with rigid constraints and the two tyres are connected to the robot using revolute motors in order to make sure that they can rotate during self-balancing.

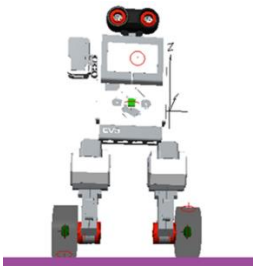

Figure 4. The complete EV3 robot in SimWise 4D.

3. Interval Type-2 Fuzzy Logic Controller (IT2FLC)

The functional block diagrams for the self-balancing system is shown in Figure 5. It is a closed loop controller where the change of output will be fed to input so that the error can be corrected automatically to produce the desired output.

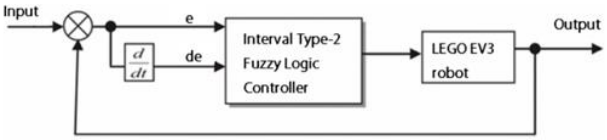

Figure 5. Functional Block Diagram for self-balancing system.

The reference input of the control system is a constant which is the original tilt angle of LEGO EV3 robot in the upright position, zero radians. Three gains are inserted into the controller act as amplification of the controller output signals. A pre-defined block is used to represent the LEGO EV3 robot in SimWise 4D system. The scope block is used to show the graph of input-output state response and the To Workspace block is used to save the data of the signal to the

workspace. Type-2 FLS has five components which are fuzzifier, rule base, inference system, type-reducer and defuzzifier as shown in Figure 6.

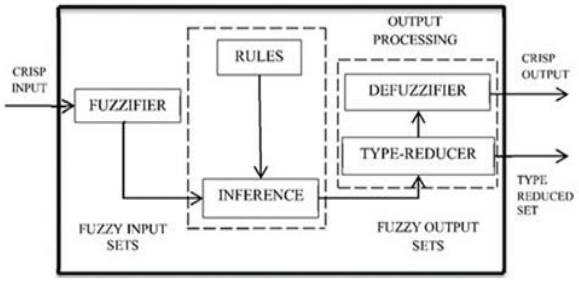

Figure 6. General Structure of IT2FLS [25].

The type-2 fuzzy set has fuzzy grades of membership compared to the type-1 fuzzy set that is crisp grades of membership. IT2FLS is an extension of T1FLS. Moreover, it has additional design degrees of freedom in FLSs.

IT2FLS has no single value for membership function but a few of it. It is like blurring type-1 membership function. Its membership function ranges over fuzzy sets of type-1. The type of reduction (Nie-Tan) is chosen to convert type-2 fuzzy sets to type-1 fuzzy sets. Try and error method is used to tune the gain to self-balance the robot.

Nie-Tan has the ability to give a closed form expression for IT2FLS output and reduce the order of IT2FLS. The Nie-Tan method produced type-1 fuzzy sets output which is the upper and lower membership functions of type-2 fuzzy sets output. By this, the x-coordinate of the geometry centroid of the footprint of uncertainty can be found to form final system output.

The expert knowledge of fuzzifier consisting of a set of fuzzy IF-THEN rules. There are 5 fuzzy levels which formed 25 rules.

Table 2. Fuzzy Rules

Error, E	Change of error, ΔE				
	NB	NS	Z	PS	PB
NB	PB	PB	PB	PS	Z
NS	PB	PB	PS	Z	NS
Z	PB	PS	Z	NS	NB
PS	PS	Z	NS	NB	NB
PB	Z	NS	NB	NB	NB

4. Particle Swarm Optimization (PSO)

PSO is a popular nature-inspired metaheuristic optimization algorithm that has emerged as a promising algorithm for solving various optimization in various field especially science and engineering. There are two script files required for full implementation of the PSO. The first file is to define the function and the second file is to develop main PSO program. The PSO is easy to implement and fewer parameters adjustment. In PSO, every single solution is a particle in the search space. All particles have fitness values that are evaluated by the fitness function for optimization. The velocities of these particles direct the flight of the particles. The particles fly through the problem space according to the current best particles [22].

PSO is initialized with a set of random particles (solutions) and then searches for the optimum solution by updating the generation. In each iteration, each particle is updated based on the two best values. The first is the best solution (fitness) achieved so far. The fitness value is also stored. This value is called Pbest. The particle swarm optimizer also tracks the best value obtained by any particle in the population so far. This best value is a global best and also

called as Gbest. When a particle takes part of the population as its topological neighbours, the best value is a local best, called Lbest. After finding the two best values, the particle updates its velocity and position. Each updating of Pbest and Gbest of the population, the fitness of solution is evaluated [30].

Rand () is a random number between (0, 1), c1 and c2 are learning factors and also called as acceleration factor. Usually, the total summation of c1 and c2 must be equal to 4. The parameters of PSO algorithm are considered as follows:

i.	Inertial weight (wmax and wmin)	: 0.9 to 0.4
ii.	Acceleration factors (c1 and c2)	: 1 and 3.
iii.	Population size, n	: 10
iv.	Maximum iteration (bird_setp)	: 100
v.	Number of Dimensions (dim)	: 5
vi.	Maximum number of runs (runMax)	: 5

The implementation of PSO is used to tune the IT2FLC function. Thus, there are five gains are considered in this PSO IT2FLS which three of them are controller gain and the other two are IT2FLC function's gain which represented as standard deviation and mean in order to determine the footprint of uncertainty (FOU). These gains are *K4* and *K5* and added in this PSO IT2FLS function only.

5. Simulation Results

The intuitive method is used during gain tuning in functional block diagram for self-balancing system. Table 3 showed the gain values of functional block diagram in each comparison for Type-1 fuzzy logic and IT2-FLC with PSO. It shows that gain *K4* and *K5* is not used for IT2FLS and only used in PSO IT2FLS. Figure 7 shows corresponding performance in terms of tilt angle and torque.

Table 3. Gain Values of Self-Balancing System.

Gain Value	IT2FLS	PSO IT2FLS
K1	0.048	0.0457
K2	0.24	0.2435
K3	100	100.1594
K4	/	0.955
K5	/	0.158

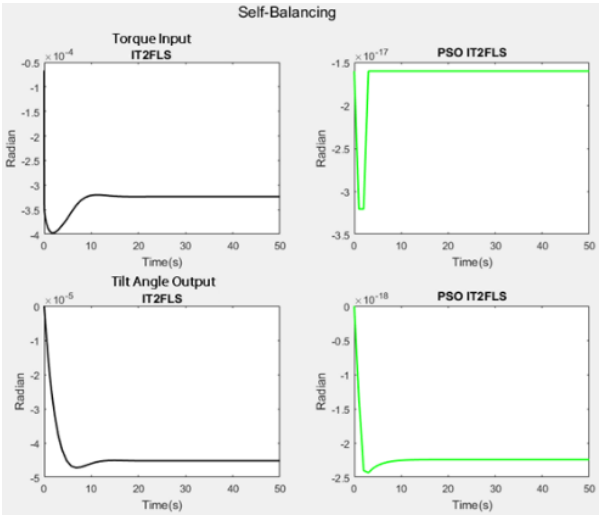

Figure 7. Graph of input and output response before and after optimised.

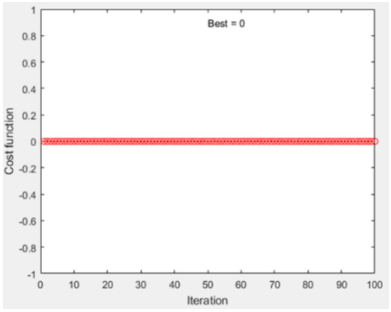

Figure 8. Evaluated fitness graph of solutions.

Table 4. State Response Before And After Optimised.

	Self-Balancing	
Time	IT2FLS	PSO IT2FLS
Rise Time, s	3.575	1.625
Peak Time, s	6.796	2.98
Settling Time, s	9.774	6.08
Overshoot, %OS	5.176	8.233
Steady State	-4.521E-05	-2.247E-18

Figure 8 shows the evaluated fitness of solutions with low dispersion during PSO. The zero error is obtained in the converged graph clearly showed the best fit for those solutions. The state response pattern of two graphs is same although the system performance is different. The PSO IT2FLS has the shortest rise time, peak time and settling time compared with IT2FLC systems. TheIT2FLS has longer rise time and settling time compared to optimised IT2FLS. Although the overshoot of PSO IT2FLS is higher than no-optimised, it is still in acceptable range and its steady state tilt angle achieved is nearer to zero radians. Hence, tuning this IT2FLC with optimization algorithms have provided a better result as can be seen in Table 4.

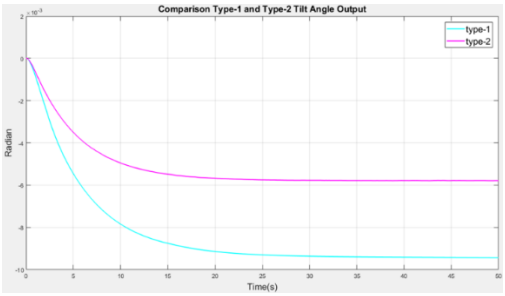

Figure 9. Comparison IT1FLS and IT2FLS Tilt Angle Output.

Table 5. Comparison between of IT1FLC and IT2FLC.

	Self-Balancing	
Time	IT1FLS	IT2FLS
Rise Time, s	12.96	11.78
Peak Time, s	/	/
Settling Time, s	22.46	19.6
Overshoot, %OS	/	/
Steady State	-9.4325E-03	-5.785E-03

In optimization solver for automatic control, the range of linear algebra problems that need to be solved on the same hardware is so varied that it is not possible to assign word-lengths based on simulation in a practical manner. Since it is impossible for PSO IT2FLS to implement in real LEGO EV3 robot. Hence, the comparison of Type-1 Fuzzy Logic System and Interval Type-2 Fuzzy Logic System is shown in Figure 9 and the respective performance is tabulated in Table 5. Peak time and overshoot are not appear in this tilt angle performance. The steady state of IT2FLS is more approximate to zero. The state response showed clearly that IT2FLS achieve steady state faster than IT1FLS. Hence, IT2FLS can self-balance the robot faster in an upright position.

6. Conclusion

In the nutshell, the objective and scope of the project are achieved and the results show that LEGO EV3 robot can self-balancing successfully using IT2FLC. The results show the value of steady state that is approximately zero which proved the efficiency of intelligent control in the self-balancing field. The Particle Swarm Optimization (PSO) also being proven is able to optimised Interval Type-2 Fuzzy Logic Controller (IT2FLS) to get a better performance in self-balancing. For disturbance rejection, Interval Type-2 Fuzzy Logic Controller (IT2FLS) gives good performance in state response. The comparison of the results also proved the robustness of Interval Type-2 Fuzzy Logic Controller (IT2FLS). For future recommendation, the system can be implemented in practical manner like EV3 robot hardware to investigate its performance.

Acknowledgments

The work presented in this paper is supported by Research grant (RDU170502) from Faculty of Electrical and Electronics Engineering, Universiti Malaysia Pahang and Ministry of Energy, Science, Technology, Environment and Climate Change, MESTECC.

References

1. T. Feng, T. Liu, X. Wang, Z. Xu, M. Zhang, and S. C. Han, "Modeling and implementation of two-wheel self-balancing robot equipped with supporting arms," *Proc. 2011 6th IEEE Conf. Ind. Electron. Appl. ICIEA 2011*, no. d, pp. 713–718, 2011.M.
2. H. S. Juang and K. Y. Lurrr, "Design and control of a two-wheel self-balancing robot using the arduino microcontroller board," *IEEE Int. Conf. Control Autom. ICCA*, pp. 634–639, 2013.
3. J. Wu and W. Zhang, "Design of fuzzy logic controller for two-wheeled self-balancing robot," *Proc. 6th Int. Forum Strateg. Technol. IFOST 2011*, vol. 2, pp. 1266–1270, 2011.
4. P. T. Behera and S. J. Mija, "Balancing of two wheeled inverted pendulum using SOSMC and validation on LEGO EV3," *1st IEEE Int. Conf. Power Electron. Intell. Control Energy Syst. ICPEICES 2016*, pp. 1–6, 2017.
5. D. Bździuch and W. Grzegozek, "A Two-Wheeled, Self-Balancing Electric Vehicle Used As an Environmentally Friendly Individual Means of Transport," *IOP Conf. Ser. Mater. Sci. Eng.*, vol. 148, no. 1, 2016.
6. R. S. Martins and F. Nunes, "Control system for a self-balancing robot," no. Project I, pp. 1–10, 2017.
7. M. R. Hao, N. M. A. Ghani, and M. S. A. Wahid, "Real-Time PID Control of Wireless Two-Wheeled Balancing Lego EV3 Robot," vol. 10, no. 1, pp. 87–92.
8. I. Mateşică, M. Nicolae, L. Bărbulescu, and A. M. Mărgeruseanu, "Self-balancing robot implementing the inverted pendulum concept," *Netw. Educ. Res. RoEduNet Int. Conf. 15th Ed. RoEduNet 2016 - Proc.*, 2016.
9. Qiu *et al.*, "Two-wheeled self-balancing robot modeling and nonlinear control," *2017 14th Int. Conf. Ubiquitous Robot. Ambient Intell. URAI 2017*, pp. 734–739, 2017.

10 S. M. I. Rumi, M. F. Hossain, I. S. M. S. Islam, and M. K. Rahman, “System design of two wheeler self-balanced vehicle,” *10th Fr. Congr. 8th Eur. Congr. Mecatronics, MECATRONICS 2014*, pp. 331–336, 2014.
11. C. Paper, “‘ An Experiment of Two-wheeled Self- Balancing Scooter ,’” no. December 2016, 2011.
12. N. N. Son and H. P. H. Anh, “Adaptive backstepping self-balancing control of a two-wheel electric scooter,” *Int. J. Adv. Robot. Syst.*, vol. 11, pp. 1–11, 2014.
13. A. D. M. Africa, “A Mathematical Fuzzy Logic Control Systems Model Using Rough Set Theory for Robot Applications,” vol. 9, no. 2, pp. 8–12.
14. C. Yang and T. Murakami, “A study on self-balancing electric motorcycles with two-wheel steering,” *2014 7th Int. Conf. Inf. Autom. Sustain. "Sharpening Futur. with Sustain. Technol. ICIAfS 2014*, 2014.
15. O. Jamil, M. Jamil, Y. Ayaz, and K. Ahmad, “Modeling, control of a two-wheeled self-balancing robot,” *2014 Int. Conf. Robot. Emerg. Allied Technol. Eng. iCREATE 2014 - Proc.*, pp. 191–199, 2014.
16. G. Yu, “PSO-based Fuzzy Control of a Self-Balancing Two-Wheeled Robot,” pp. 1–5, 2017.
17. A. Zeghoudi and A. Chermitti, “A Comparison between a Fuzzy and PID Controller for Universal Motor,” vol. 104, no. 6, pp. 32–36, 2014.
18. J. L. Jerez, “Custom Optimization Algorithms for Efficient Hardware Implementation,” *Thesis*, no. May, 2013.
19. Jinkun Liu, *Intelligent Control Design and MATLAB Simulation*. Beijing, China: Springer, Singapore, 2018.
20. E. C. Daniel Zaldier, “An Educational Fuzzy-based Control Platform using LEGO Robots,” *Int. J. Electr. Eng. Educ. Manchester Univ. Press*, vol. 50, no. 2, pp. 157–171, 2013.
21. D. Liang, N. Sun, Y. Wu, and Y. Fang, “Modeling and motion control of self-balance robots on the slope,” *Proc. - 2016 31st Youth Acad. Annu. Conf. Chinese Assoc. Autom. YAC 2016*, pp. 93–98, 2017.
22. M. A. Haraoubia, N. Essounbouli, and A. Hamzaoui, “Wind turbine system optimisation using interval T2FL tuned with PSO,” *IFAC-PapersOnLine*, vol. 49, no. 12, pp. 680–685, 2016.
23. Baghaeian and A. A. Akbari, “Adaptive interval type-2 fuzzy logic systems for vehicle handling enhancement by new nonlinear model of variable geometry suspension system,” *J. Vibroengineering*, vol. 19, no. 6, pp. 4498–4515, 2017.
24. S. M. Abuelenin and R. F. Abdel-Kader, “Closed-Form Mathematical Representations of Interval Type-2 Fuzzy Logic Systems,” 2017.
25. J. R. Castro, “Interval Type-2 Fuzzy Logic for Intelligent Control,” pp. 592–597, 2007.
26. M. Almaraashi, G. S. Member, R. John, S. Member, and S. Coupland, “Designing Generalised Type-2 Fuzzy Logic Systems using Interval Type-2 Fuzzy Logic Systems and Simulated Annealing,” pp. 10–15, 2012.
27. J. M. Mendel, R. I. John, and F. Liu, “Interval Type-2 Fuzzy Logic Systems Made Simple,” *Fuzzy Syst. IEEE Trans.*, vol. 14, no. 6, pp. 808–821, 2006.
28. B. Panomruttanarug and P. Chotikunnan, “Self-balancing iBOT-like wheelchair based on type-1 and interval type-2 fuzzy control,” *2014 11th Int. Conf. Electr. Eng. Comput. Telecommun. Inf. Technol. ECTI-CON 2014*, 2014.
29. M. Ri, J. Huang, S. Ri, H. Yun, and C. Kim, “Design of interval type-2 fuzzy logic controller for mobile wheeled inverted pendulum,” *2016 12th World Congr. Intell. Control Autom.*, pp. 535–540, 2016.
30. M. N. Alam, “Particle Swarm Optimization : Algorithm and its Codes in MATLAB,” no. March, pp. 1–10, 2016.